Scripting for Sys Admins

Session 618

These are confidential sessions—please refrain from streaming, blogging, or taking pictures.

Auto-enrolling with an MS CA

Making magic happen with certificates

Joel Rennich
Consulting Engineering Manager
US Commercial and Government Sales
Apple

3G
9:42 AM
Messages
Tuesday
9
Calendar
Photos
Camera
YouTube
Stocks
280
Maps
73°
Weather
Voice Memos
Notes
Clock
Calculator
Settings
iTunes
App Store
Compass
Phone
Mail
Safari
iPod

Enterprise Identities

- Certificates used for a variety of authentication uses
  - VPNs
  - 802.1X
  - Websites
- Mac OS X has a rather robust certificate implementation, but…
  - Can require user interaction
  - No method for automatically enrolling

How the Other Half Lives

- MS provides a certificate authority (CA)
- MS clients respond to a policy to auto-enroll
- Client systems generate the CSR, get it signed and then import the resulting identity into a local store
- Done as the machine, not necessarily as the user
- Most users never even know…

A Quick Tour of the MS CA

Let's see what we have to work with…

MS CA

- Web-based
- Works with IIS, so you have to turn that on
- Kerb auth

I have no idea how this actually works on a PC

Reversing the Engineering

Sneaking in over http

Credentials

- Who you log in as determines what you get
- All AD users can get kerb creds
- All AD-bound machines have a machine account that works as well

Log on to the web CA

Use Safari and tcpdump to find out the transaction

Bing! for it

Alpha for it

WolframAlpha™ computational knowledge engine

autoenroll certificate from MS web CA

Wolfram|Alpha isn't sure what to do with your input. Tips for good results »

More to explore:

» Astronomy
» Music
» Organizations
» Food & Nutrition
» Earth Sciences
» Engineering
» Socioeconomic Data
» Web & Computer Systems
» More...

Are you an expert on this topic? Find out how you can help. »

Google for it

Tell someone who works for you to do it

The Script

Credentialling

- `kinit -k` get a ticket from the local keytab as root
  - Allows to auth as the machine without having to read in the AD plist and reverse the password
  - For 2k8 environments—use Snow Leopard

Nonexportable Certs

- `security -x`
  - Imports identity and prevents it from being exported
  - New to Snow Leopard

Automated Image Development

Josh Wisenbaker
Consulting Engineer
US Commercial and Government Sales
Apple

Introduction

What are we talking about?

- Making and testing images
- Cutting steps with automation

Traditional Imaging Process

One step at a time…

- Make an image
- Make a NetInstall ASR set
- Deploy the image
- Test the image

Traditional Imaging Process

Make an image

- Install Mac OS X
- Install apps
- Tweak the settings
- Cleanup the system
- FireWire target disk mode

Traditional Imaging Process

Make a NetInstall ASR set

- Move image to server
- Create System Image Utility workflow
- Run workflow

Traditional Imaging Process

Deploy the image

- Enable NetInstall image in server admin
- Set client Mac to boot from the correct NetInstall set
- Image client

11 Steps

Rinse and repeat

Traditional Imaging Process

This is WWDC after all…

```
10 Make an image
20 Make a NetInstall ASR set
30 Deploy the image
40 Test the image
50 goto 10
60 goto home
```

Traditional Imaging Process

This is WWDC after all…

```
10 Make an image
20 Make a NetInstall ASR set
30 Deploy the image
40 Test the image
50 goto 10
60 goto home
```

“I’m getting too old for this.”

Josh Wisenbaker
Me

What Do I Want to Do?

Less Work!

Josh, I need to you work both faster and harder.

All request for less work are denied.

What would @fake michael_bay do?

WWFMBD?

Bring the pain

- Get a latte
- Use some robots
- BOOM!

Get Me a Latte

sudo grep latte /dev/drinks>me

- I've got a mentee at work
- I made him get me a latte
- I live in NC
- He lives in NYC
- BOOM!

Automatons

Image creation automation

- On Snow Leopard use SIU
  - Create NetRestore set from Installer media
- On Leopard use InstaDMG
  - Create ASR image from Installer media
- BOOM!

More Robots

Image deployment automation

- Feed deployment images to SIU
  - Automator workflow
    - Create automated deployment set
    - Set MAC filter to just the test Mac

```
automator create_NetResotre.workflow
```

- BOOM!

Robot Fight!

Image deployment automation

- `scp` .nbi set to the NetBoot Server
  - Enable image in the NBImageInfo.plist
  - Reset the MAC filter on the image
  - `/usr/libexec/PlistBuddy`
- Set client Mac to boot from NetInstall set
  - `bless --netboot --server bsdp://en1@17.203.12.203`
  - `restart -r now`
- BOOM!

Autonomous Automation

Let launchd do your bidding

- Take all these steps and just schedule them
- Runs while you sleep
- BOOM!

Boom!

Boom!

Less work for me!

- Image creation is automated
- NetInstall creation is automated
- Image deployment is automated

Wrapping Up…

You’re wrapping up already? That was like 6 minutes.

The WWDC people will not be happy.

Bonus Solution!

Automated Local Management Policy

Introduction

What are we talking about?

- Creating local management policy
- Scripting the deployment of that policy

Local Management Policy

Why would I do that?

- Can’t create computer records in DS
- Can’t extend DS schema

Local Management Policy

It's a setup!

- Create local computer object
- Set policy with Workgroup Manager
- Export policy with `dscl -mcxexport`

Local Management Policy

Suck it up!

- Create local computer object
- Import policy with `dscl -mcximport`

Local Management Policy

Tools you need

- Workgroup Manager
- `dscl`
- `launchd`

Local Management Policy

Extra tools we like

- `mcxrefresh`
- Lingon

How Do I Do It?

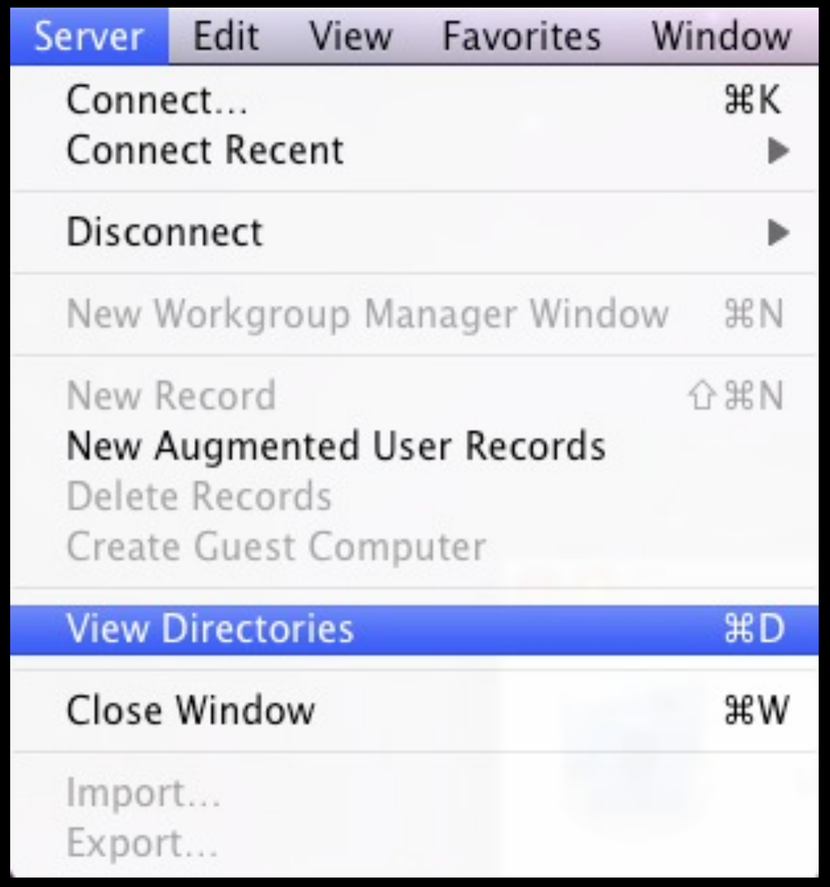
Create Account

WGM FTW!

Create Account

WGM FTW!

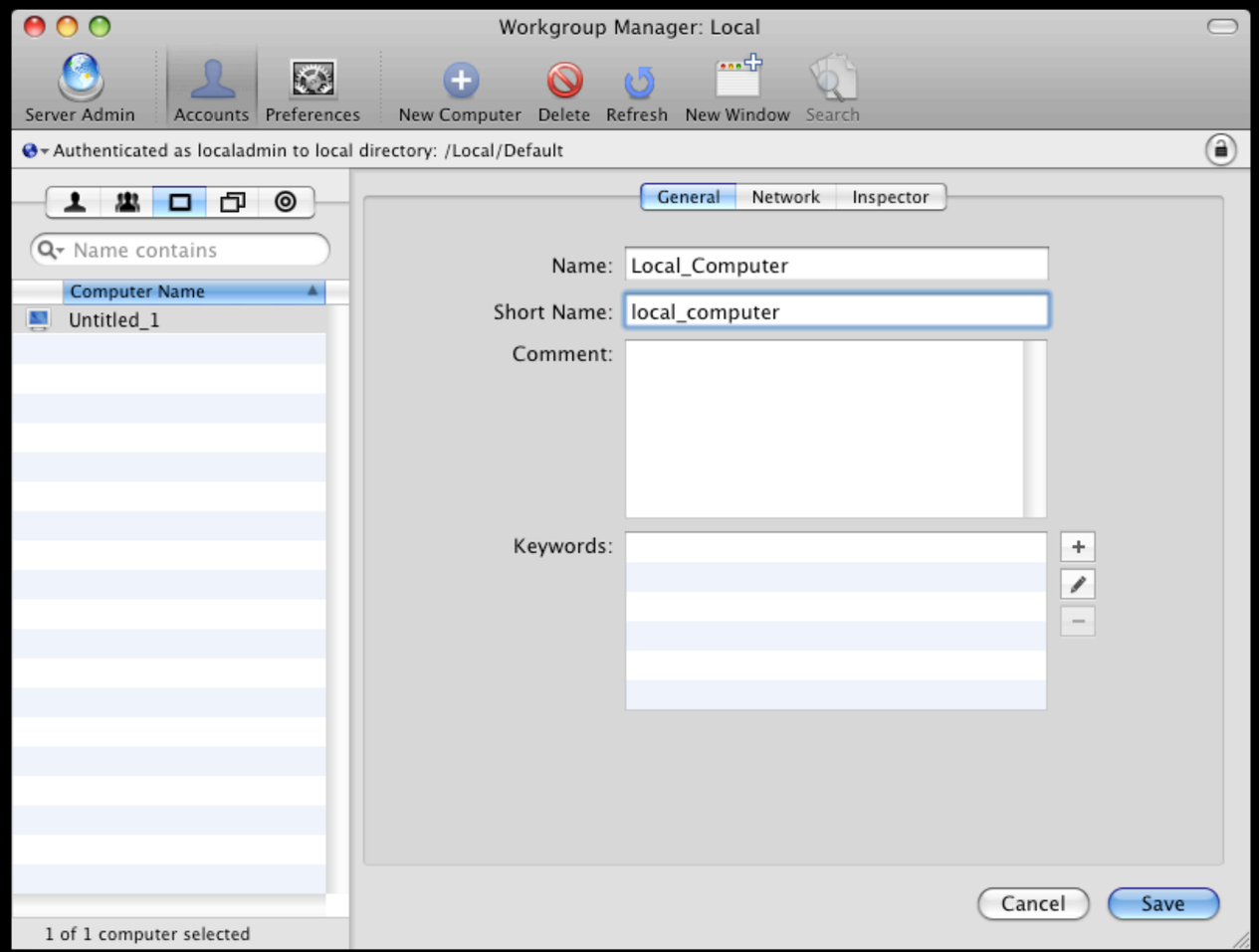
- Use Server > View Directories



Create Account

WGM FTW!

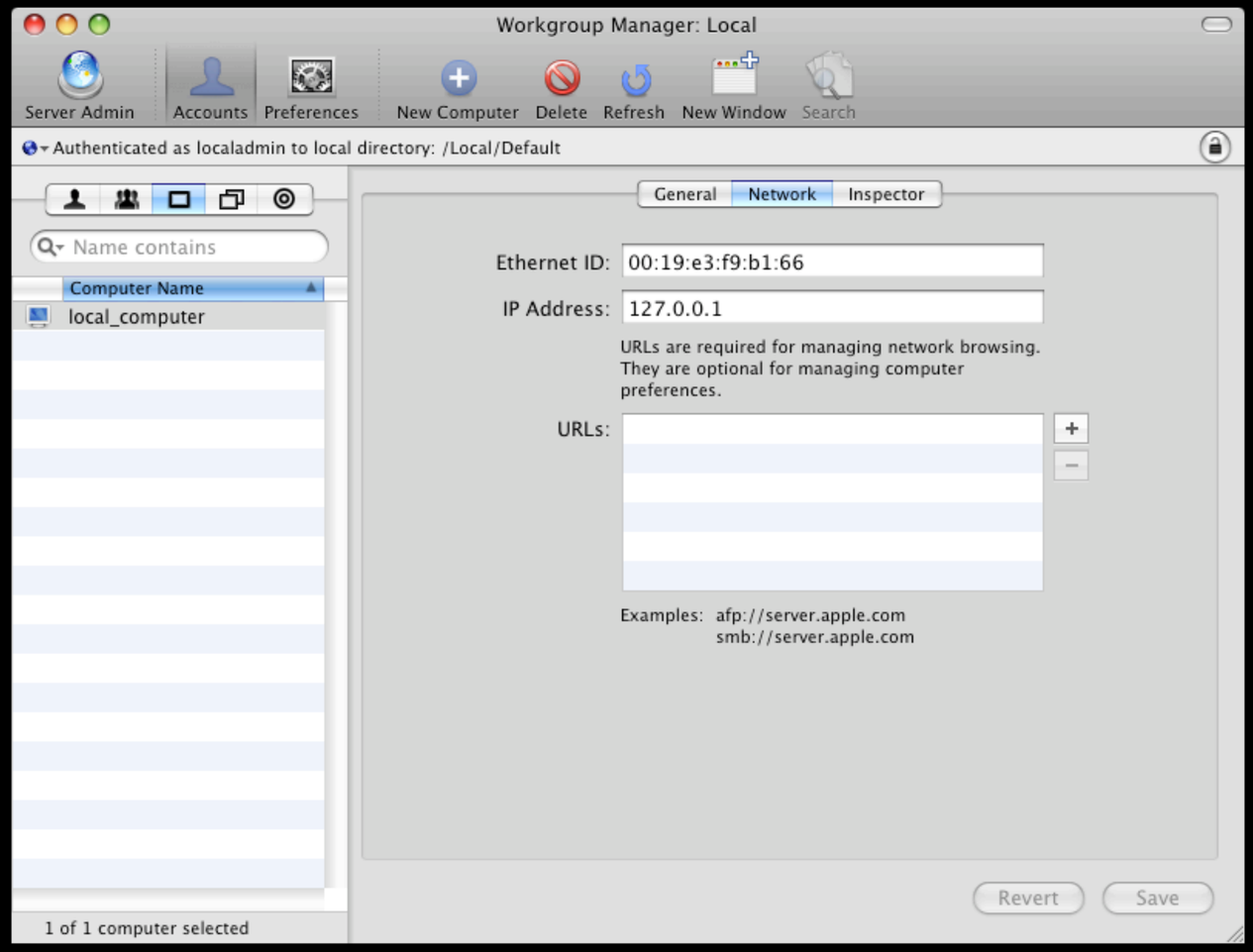
- Use Server > View Directories
- Click New Computer in /Local/Default



Create Account

WGM FTW!

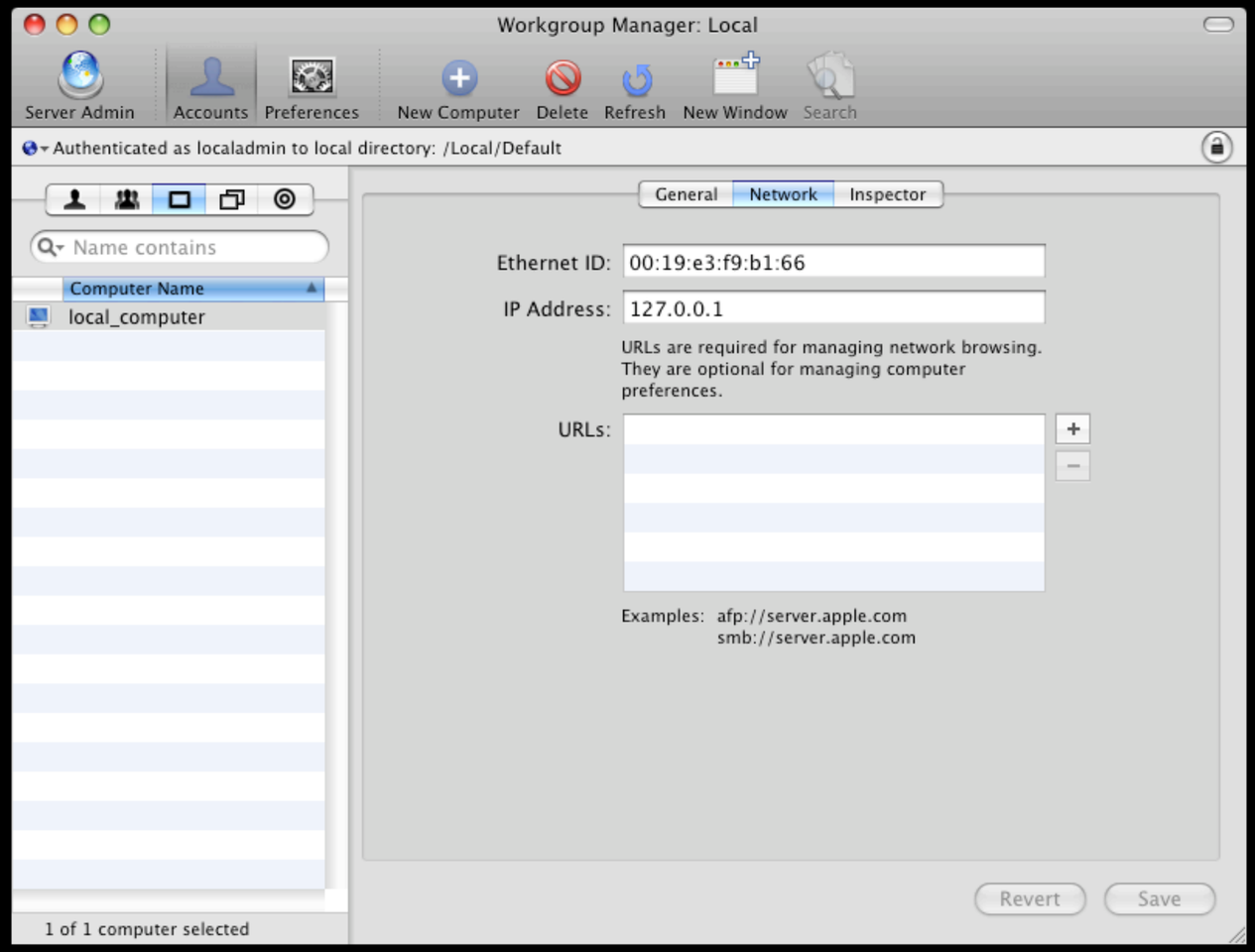
- Use Server > View Directories
- Click New Computer in /Local/Default
- Enter MAC and loopback address



Create Account

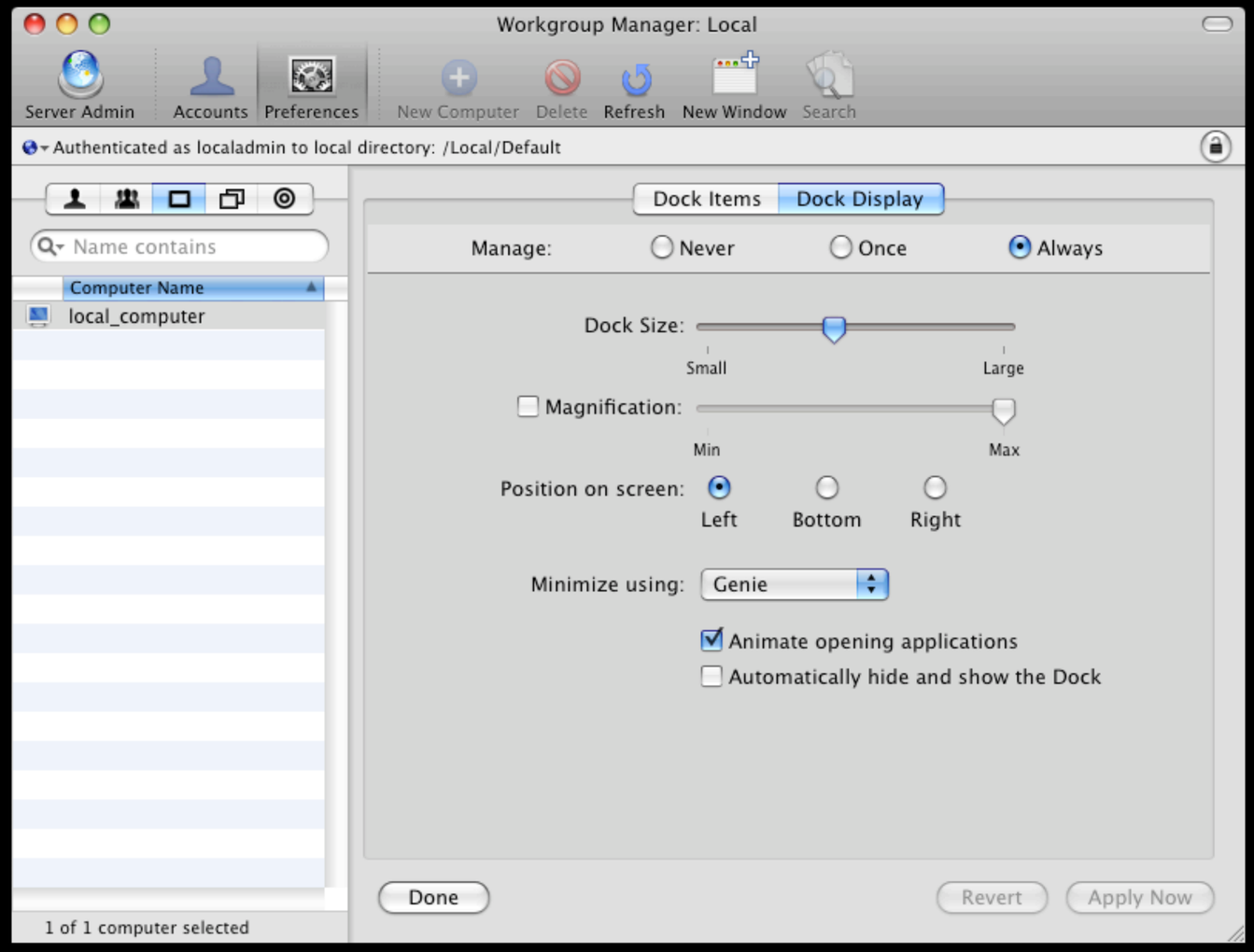
WGM FTW!

- Use Server > View Directories
- Click New Computer in /Local/Default
- Enter MAC and loopback address
- That's it!



Local Computer Object Policy

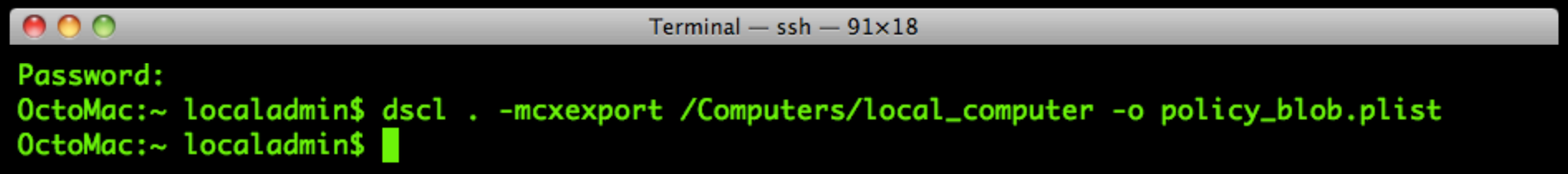
Boss your Mac around



Export Policy Blob

Get it out of here!

- `dscl -mcxexport`



Um…What Now?

Local Management Policy

Start it up!

- Create local computer object
- Import policy with `dscl -mcximport`

Create Local Account

Hi bash! Nice to see you

```
#!/bin/bash
dscl . -create /Computers/localhost
dscl . -create /Computers/localhost RealName localhost
dscl . -create /Computers/localhost ENetAddress $(ifconfig en0 \
| grep ether | awk '{print $2}')
dscl . -create /Computers/localhost IPAddress 127.0.0.1
dscl . -mcximport /Computers/localhost <pathto>policy_blob.plist
rm /Library/LaunchDaemons/com.apple.LocalMCX
exit 0
```

Create Local Account

Hi bash! Nice to see you

```
#!/bin/bash
dscl . -create /Computers/localhost
dscl . -create /Computers/localhost RealName localhost
dscl . -create /Computers/localhost ENetAddress $(ifconfig en0 \
| grep ether | awk '{print $2}')
dscl . -create /Computers/localhost IPAddress 127.0.0.1
dscl . -mcximport /Computers/localhost <pathto>policy_blob.plist
rm /Library/LaunchDaemons/com.apple.LocalMCX
exit 0
```

Create Local Account

Hi bash! Nice to see you

```
#!/bin/bash
dscl . -create /Computers/localhost
dscl . -create /Computers/localhost RealName localhost
dscl . -create /Computers/localhost ENetAddress $(ifconfig en0 \
| grep ether | awk '{print $2}')
dscl . -create /Computers/localhost IPAddress 127.0.0.1
dscl . -mcximport /Computers/localhost <pathto>policy_blob.plist
rm /Library/LaunchDaemons/com.apple.LocalMCX
exit 0
```

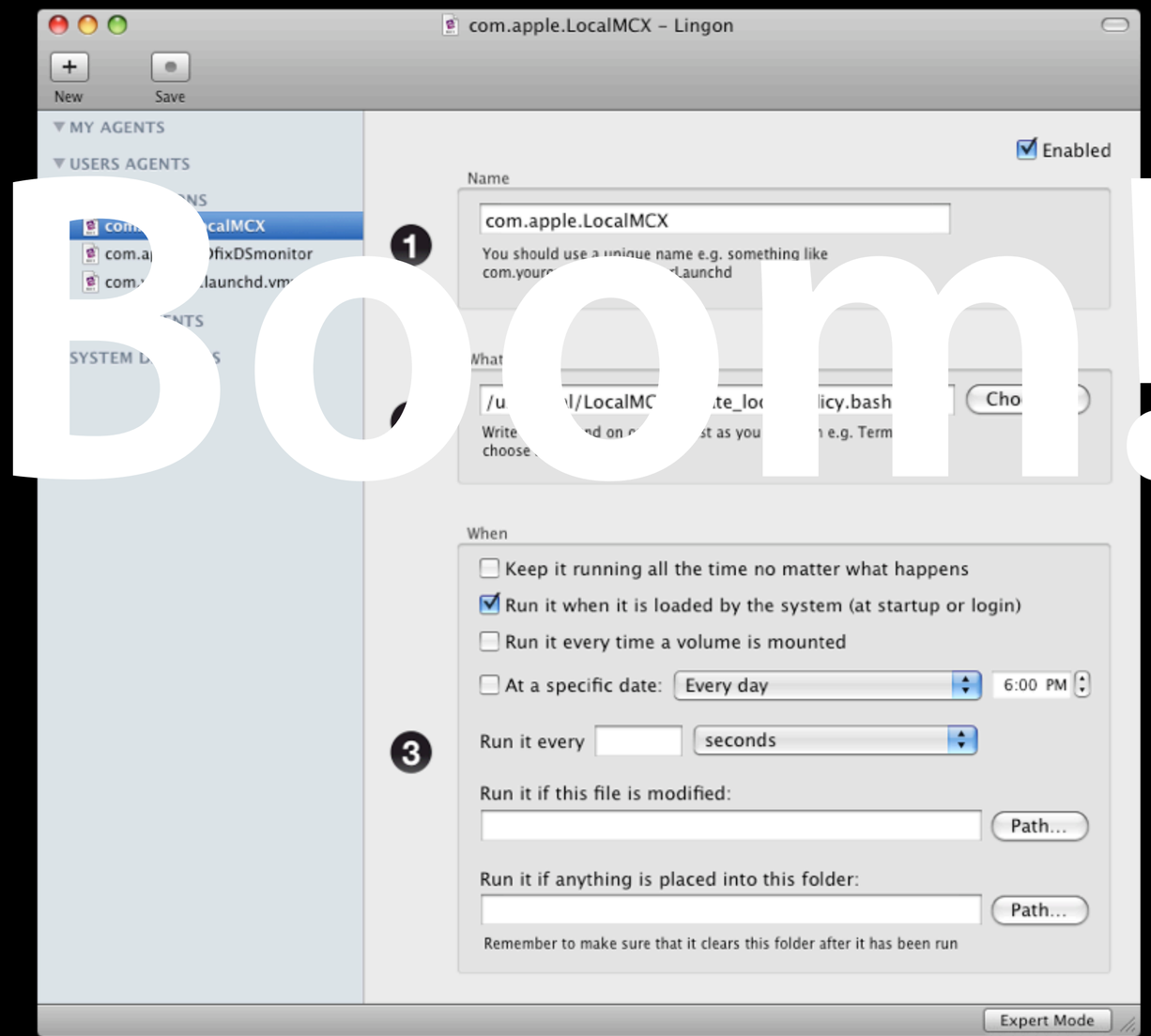
Create Local Account

Hi bash! Nice to see you

```
#!/bin/bash
dscl . -create /Computers/localhost
dscl . -create /Computers/localhost RealName localhost
dscl . -create /Computers/localhost ENetAddress $(ifconfig en0 \
| grep ether | awk '{print $2}')
dscl . -create /Computers/localhost IPAddress 127.0.0.1
dscl . -mcximport /Computers/localhost <pathto>policy_blob.plist
rm /Library/LaunchDaemons/com.apple.LocalMCX
exit 0
```

Run at Startup

launchd+Lingon=Easy



Boom!

Demo

From the Command Line to the Desktop

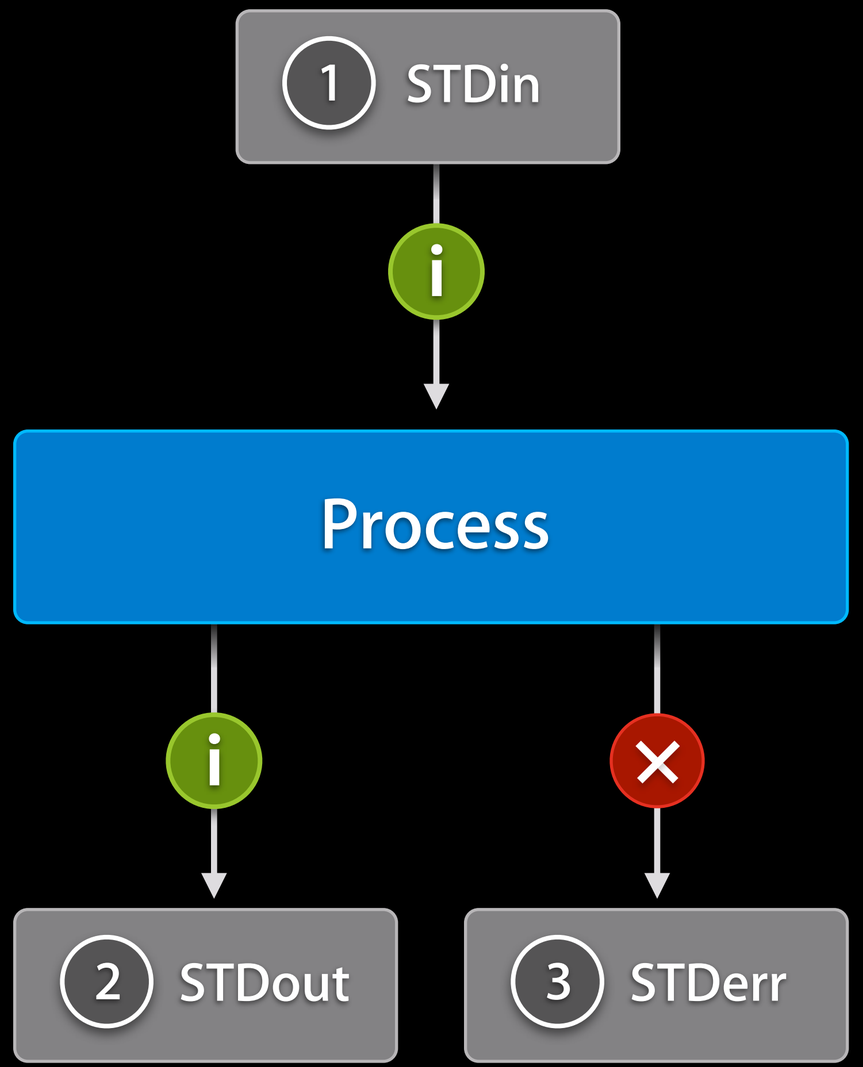
Timothy Perfitt
Consulting Engineer
US Commercial and Government Sales
Apple

The Space Between

- UNIX standard file descriptors
  - Stdin
  - Stdout
  - StdErr
- Signals
  - HUP, TERM
- The Script Interface
  - getopt

STDin, STDout, STDerr

- STDin
  - Pass info into process
- STDout
  - Pass info out of a process
- STDerr
  - Pass errors out of a process without intermixing in STDout



STDin, STDout, STDerr

```
#!/bin/bash

echo Process ID is $$

echo "this goes to stdout"
echo "this also goes to stdout" >&1
echo "this goes to stderr" >&2

while read input_line
    do echo You Entered $input_line
done
```

“Signals allow the manipulation of a process from outside its domain.”

—man signal

Signals

Signals

```
#!/bin/bash

cleanup () {
    echo caught signal
    return
}

trap cleanup HUP TERM
echo Process ID is $$

echo "this goes to stdout"
echo "this also goes to stdout" >&1
echo "this goes to stderr" >&2

while read input_line
    do echo You Entered $input_line
done
```

GetOpt

- String parsing is for suckers
- Easily parse arguments
- Provides a consistent interface
- Saves time
- Time equals money
- Money allows more free time
- Free time allows for more time to write scripts

GetOpts in Bourne

```
#!/bin/sh

# Start processing options at index 1.
while getopts "vc:" VALUE  ; do

    if [ "$VALUE" = "v" ] ; then
        echo verbose
    fi

    if [ "$VALUE" = "c" ] ; then
        echo c set to $OPTARG
    fi

done
```

GetOpt in Ruby

```
require "getopt/std"

   # Look for -o with argument, and -I and -D boolean arguments
   opt = Getopt::Std.getopts("o:ID")

   if opt["I"]
      # Do something if -I passed
   if opt["D"]
      # Do something if -D passed

   if opt["o"]
      case opt["o"]
         # blah, blah, blah
      end
   end
```

*http://rubyforge.org/docman/view.php/735/281/README.html

GetOpt in Perl

```
use Getopt::Long;
  my $data   = "file.dat";
  my $length = 24;
  my $verbose;
  $result = GetOptions ("length=i" => \$length,    # numeric
                        "file=s"   => \$data,      # string
      "verbose"  => \$verbose);  # flag
```

* from http://perldoc.perl.org/Getopt/Long.html

GetOpt in Python

```
def main():
    try:
        opts, args = getopt.getopt(sys.argv[1:], "ho:v", ["help", "output="])
    except getopt.GetoptError, err:
        # print help information and exit:
        print str(err) # will print something like "option -a not recognized"
        usage()
        sys.exit(2)
    output = None
    verbose = False
    for o, a in opts:
        if o == "-v":
            verbose = True
        elif o in ("-h", "--help"):
            usage()
            sys.exit()
        elif o in ("-o", "--output"):
            output = a
        else:
            assert False, "unhandled option"
    # ...

if __name__ == "__main__":
    main()
```

* from http://docs.python.org/library/getopt.html

GetOpt in Fortran

twitter

Login Join Twitter!

@tperfitt Fortran programmers do it with double precision.

about 15 hours ago from Tweetie in reply to tperfitt

nigelkersten

Objective-C

- NSTask
  - Create a subprocess
- NSPipe
  - Communicate with STDin, STDout, and STDerr of process
- Kill
  - Send a signal
- System
  - Pass a command to the shell
    - Runs synchronously

Objective-C Tasks

```
NSTask *convertAccountTask;

NSPipe *outPipe=[NSPipe pipe];
NSPipe *inPipe=[NSPipe pipe];

[convertAccountTask setStandardOutput:outPipe];
[convertAccountTask setStandardInput:inPipe];

NSString *launchPath=[[NSBundle mainBundle]
    pathForResource:@"mobiletransform" ofType:@"sh"];

[convertAccountTask setLaunchPath:launchPath];

[convertAccountTask setArguments:[NSArray arrayWithObjects:
    @"-v",@"-U",   [usernameTextField stringValue],@"-P",@"-o",
    [smbHome stringValue],@"-u",[[userPopup selectedItem] title],nil]];
```

Objective-C Tasks

```
NSNotificationCenter *defaultCenter=[NSNotificationCenter defaultCenter];

[defaultCenter addObserver:self selector:@selector(convertCompleted:)
    name:NSTaskDidTerminateNotification object:nil];

[convertAccountTask launch];
```

Objective-C Notification

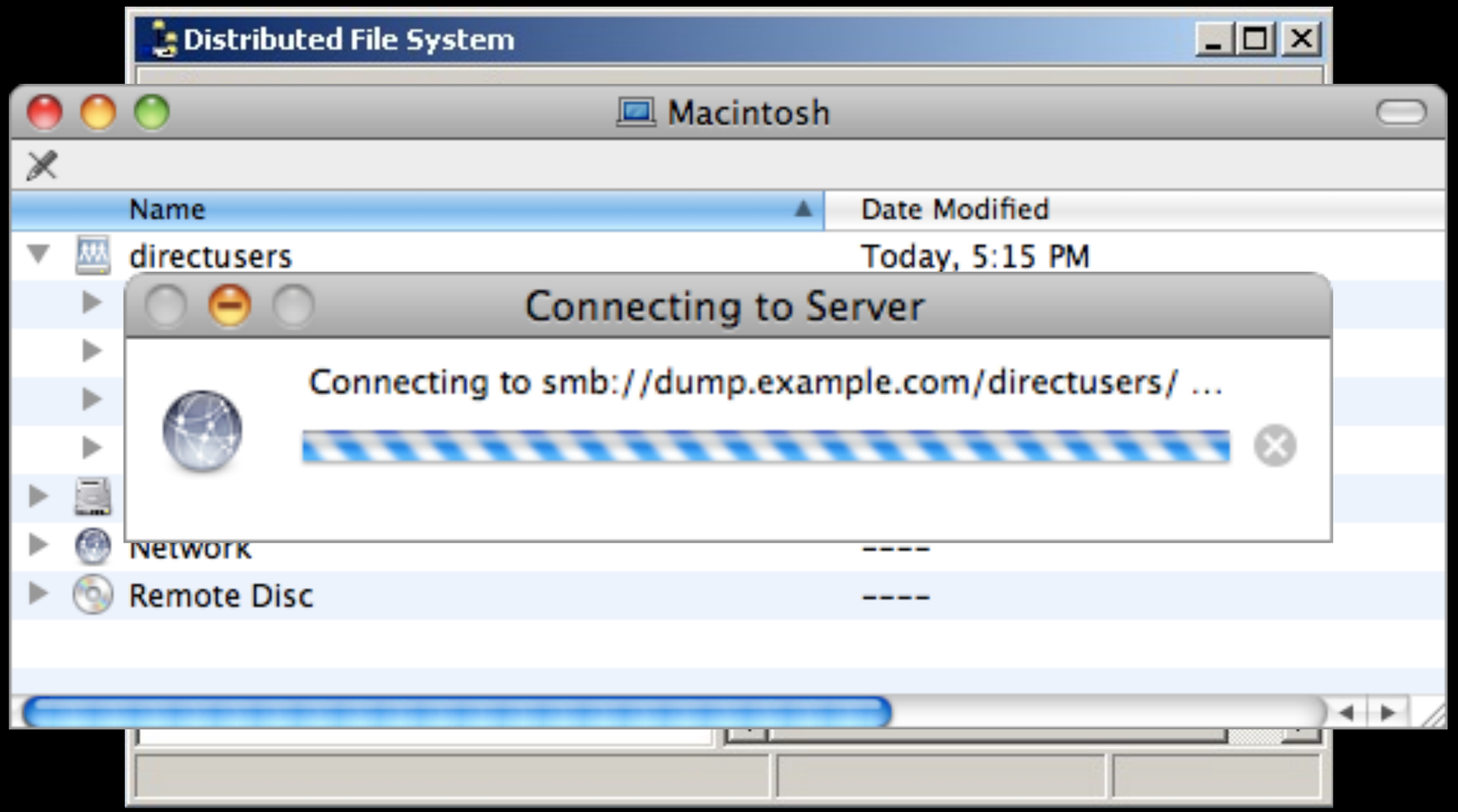
```
-(void)convertCompleted:(NSNotification *)incoming{

    NSNotificationCenter *defaultCenter=[NSNotificationCenter defaultCenter];
    [defaultCenter removeObserver:self];
    NSData *inData = nil;
    NSFileHandle *readHandle = [outPipe fileHandleForReading];

    while ((inData = [readHandle availableData]) && [inData length]) {
        NSString *oldLogText=[logTextView string];
        NSString *incomingLogText=[[NSString alloc] initWithData:inData
encoding:NSASCIIStringEncoding];
        NSMutableString *newLogText=[NSMutableString string];
        if ([oldLogText length]>0) {
            [newLogText insertString:oldLogText atIndex:0];
        }
        [newLogText insertString:incomingLogText atIndex:[newLogText length]];
        [logTextView setString:newLogText];
    }
    [convertAccountTask release];

}
```

Examples

DFS Mounter

Mobile Creator

Problem->DFS


Distributed File System
Macintosh
Name
Date Modified
directusers
Today, 5:15 PM
Connecting to Server
Connecting to smb://dump.example.com/directusers/ ...
Network
Remote Disc

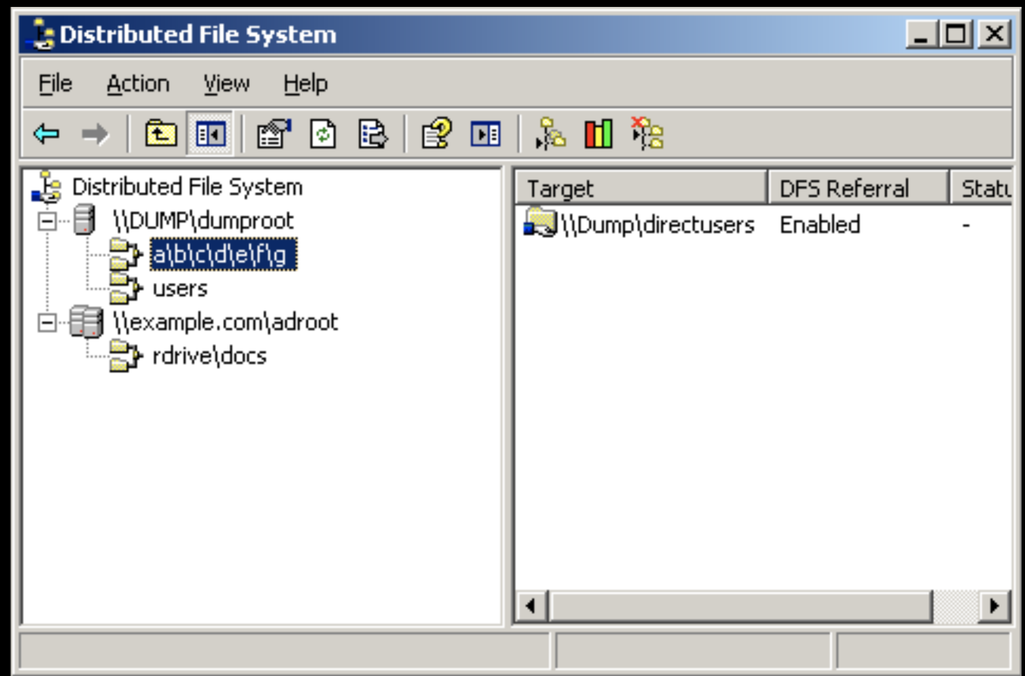
Solution==jCIFS

tperfitt$ java -cp .:jcifs-krb5-1.3.1.jar KerberosAuthExample \

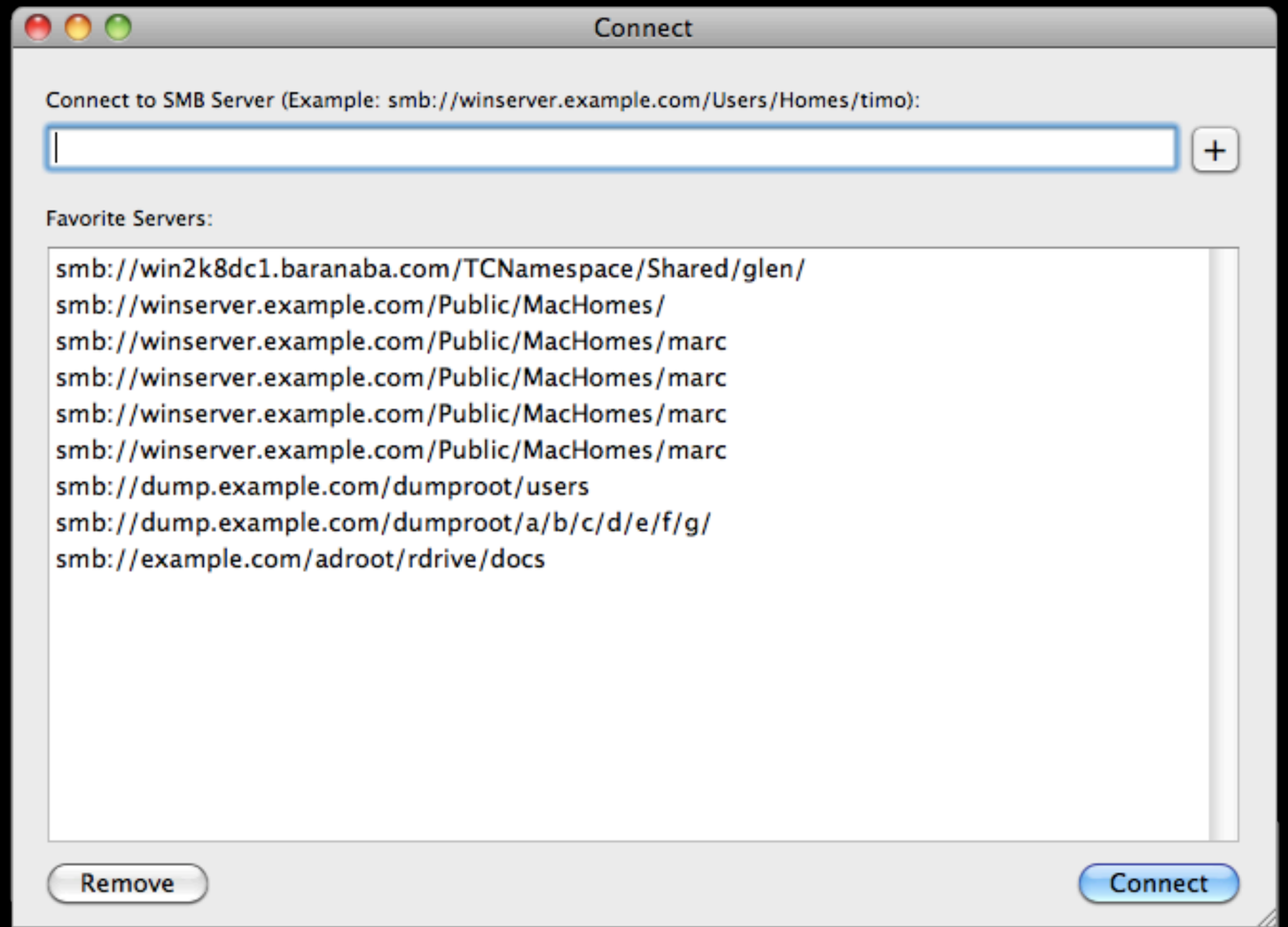
> smb://dump.example.com/dumproot/a/b/c/d/e/f/g/ \

> dump.example.com:88 EXAMPLE.COM

smb://Dump/directusers//
tperfitt$



DFS Mounter


Connect
Connect to SMB Server (Example: smb://winserver.example.com/Users/Homes/timo):
+
Favorite Servers:
smb://win2k8dc1.baranaba.com/TCNamespace/Shared/glen/
smb://winserver.example.com/Public/MacHomes/
smb://winserver.example.com/Public/MacHomes/marc
smb://winserver.example.com/Public/MacHomes/marc
smb://winserver.example.com/Public/MacHomes/marc
smb://winserver.example.com/Public/MacHomes/marc
smb://winserver.example.com/Public/MacHomes/marc
smb://dump.example.com/dumproot/users
smb://dump.example.com/dumproot/a/b/c/d/e/f/g/
smb://example.com/adroot/rdrive/docs
Remove
Connect

DFS Mounter

```
#!/usr/bin/perl -w

use Getopt::Long;


$SIG{'TERM'} = 'catch';
$SIG{'INT'} = 'catch';

sub catch {
    printf("caught signal, kill child processes and dying\n");
    kill TERM => -$$;
    #  cleanup;
    exit(-1);
}
```

DFS Mounter

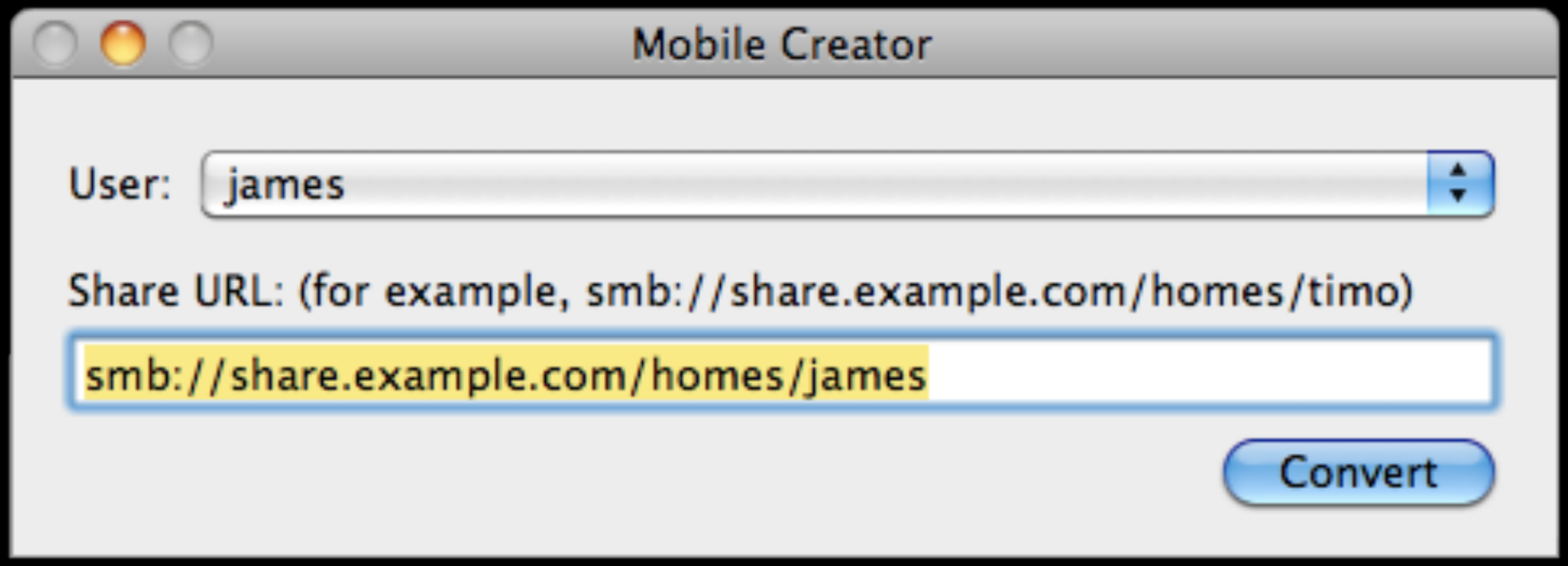
```
my $kdc   = "";
my $realm = "";
my $url="";

GetOptions ("kdc=s" => \$kdc,"realm=s"=> \$realm,"url=s"  => \$url);

if (($kdc eq "") || ($realm eq "" ) || ($url eq "")) {
    print "You must specify a realm, kdc, and url\n";
    exit -1;
}

my $resolvedURL=`/usr/bin/java -cp .:jcifs-krb5-1.3.1.jar ResolveDFSURL
$url $kdc $realm`;
```

Problem->Mobile Account Without DS

Solution==Josh+dscl

- Add to <user>.plist in local DS:
  - smb_home
  - original_home_loc
  - copy_timestamp
  - preserved_attributes
- Change authentication_authority
  - <string>;LocalCachedUser;::</string>
- Same password between SMB server and local account
  - Allows Login and Logout Sync to work

Screenshot of Mobile Creator



Mobile Account Without DS Script

```
usage () {
echo Usage:
   echo $0  '-U <local admin username> -u <username> -P -v -o <smbhome URI>'
   return
}
```

```
  trap cleanup HUP TERM
```

```
cleanup () {
   echo `date`:Operation cancelled.
   exit
}
```

```
trap cleanup HUP TERM
```

```
   while getopts "U:Pvu:o:" VALUE  ; do

   if [ $VALUE == "v" ]; then
      verbose=1
```

Mobile Account Without DS Script

```
while getopts "U:Pvu:o:" VALUE  ; do

 if [ $VALUE == "v" ]; then
    verbose=1


 elif [ $VALUE == "u" ]; then
    username=$OPTARG

    elif [ $VALUE == "o" ]; then
       originalhome=$OPTARG


     elif [ $VALUE == "U" ]; then
       authusername=$OPTARG


     elif [ $VALUE == "P" ]; then
       read authpassword
     fi
done
```

Mobile Account Without DS Script

```
dscl -u $authusername -P $authpassword . create /Users/${username}\
     authentication_authority ";LocalCachedUser;::"

if [ $verbose -eq 1 ]; then echo "`date`:setting timestamp"; fi
currentdate=$(TZ=Z date "+%Y-%m-%dT%H:%M:%SZ")

dscl -u $authusername -P $authpassword . create /Users/$username copy_timestamp \
     "$currentdate"

if [ $verbose -eq 1 ]; then echo "`date`:setting dsAttrTypeStandard:RealName"; fi
dscl -u $authusername -P $authpassword . create /Users/$username preserved_attributes\
     "dsAttrTypeStandard:RealName"

if [ $verbose -eq 1 ]; then echo "`date`:setting dsAttrTypeStandard:CopyTimestamp"; fi
dscl -u $authusername -P $authpassword . merge /Users/$username \
     preserved_attributes "dsAttrTypeStandard:CopyTimestamp"

  ...
```

Demo

Please no screaming or biting

Related Sessions

System Image Creation and Deployment with Snow Leopard Server	Marina Thursday 3:30 PM
AppleScript and Cocoa Bridge	North Beach Friday 10:30AM
Secure Access with 802.1X	Nob Hill Friday 2:00 PM

Related Labs

Mac OS X System Administration Lab	Information Technologies Lab Friday 9:00 AM
Mac OS X Installer and PackageMaker Lab	OS X Lab D Friday 9:00 AM
Client Management Lab	Information Technologies Lab Friday 9:00 AM
AppleScript and Cocoa Bridge Lab	OS X Lab C Friday 2:00 PM

Q&A

WWDC Session Survey

You can rate your sessions on the WWDC Attendee Site.

Survey forms are located on each session description page at:

developer.apple.com/wwdc/attendee

WWDC09